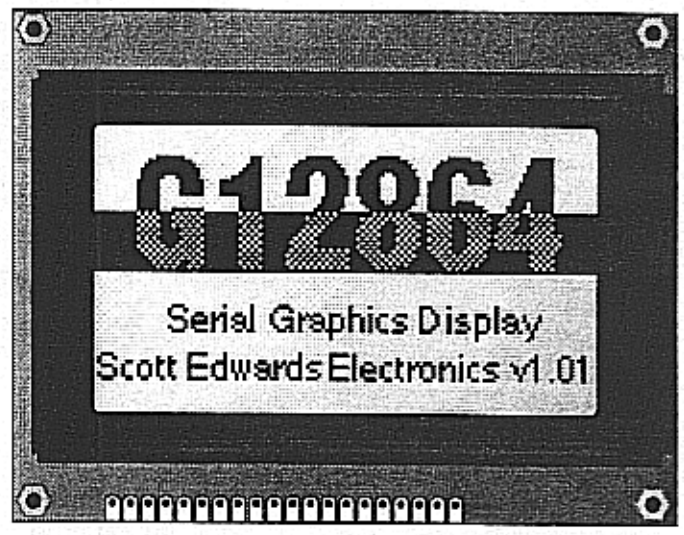
P/N 150973

Serial LCD Graphics Module (G12864)

The Serial LCD Graphics Module model G12864 receives data serially at 2400 or 9600 bits per second (bps) and displays text in a 4-line by 16-character format. The controller understands common terminal/printer control codes (such as linefeeds, carriage returns, tabs, backspace, formfeeds) for easy formatting. It extends the terminal model with additional features like backlight control, downloadable 8-by-16-pixel text font, and easy cursor positioning.

The G12864 supports graphics on its 128-by-64-pixel screen. It can plot points, draw lines, and display full-screen images from nonvolatile memory. Since text and graphics are on separate layers, the display can quickly alter one layer (e.g., clear the text screen) without affecting the contents of the other (leave graphics intact). Layers can be individually turned on and off without losing their contents and combined in ways that selectively copy, invert or mask them.

The G12864 has 16kB of nonvolatile flash EEPROM memory. The lower 2kB of this memory holds the bitmapped font used to display text. The remaining memory can be used to store up to 14 full-screen images for display on the graphics layer. A DOS utility program included with the starter package allows you to convert and download bitmapped pictures (128x64-pixel .BMP graphics) to the display.



Read Me: New-Feature Update

Version 1.1 of the G12864 firmware has two new features that are not described in the body of this manual. This firmware is identified by the number "026" at the end of the ID number on the 28-pin controller IC. Owners of earlier models can upgrade this chip for a nominal fee; contact Scott Edwards Electronics, Inc. The new features are:

• **Vertical Flip.** Configuration switch 7 (now labeled NORM/FLIP) turns the screen image upside down. This allows you to mount the display upside down in order to have the power and serial cables exit the bottom of the board. Like all configuration switches, NORM/FLIP takes effect only at power up. Remove power, change the switch, then power up.

• **Inverse-video Text.** You can now selectively invert portions of text as they are printed to the display. Control-B (ASCII 1) begins inverse-video text; Control-C (ASCII 2) ends it. Inverse video reverses the relationship of dark and light pixels; in the default setup, text is printed dark on a light background. Text printed after a Control-B is light on a dark background.

Table of Contents

Description 1
Disclaimer of Liability 3
Warranty, Return/Repair/Replacement Policy 3
Quick Checkout and Demonstration 4
Configuring the G12864 4
Connecting Power 5
Connecting Serial Input 5
Display Contrast 6
Getting Acquainted 7
Text-Layer Control Codes and Special Features 8
Graphics Layer Instructions 13
Painless Graphics Primer 18
Creating and Downloading Bitmap Fonts and Graphics 20
Design Notes and Specifications 23
Recipes and Example Programs 26

Disclaimer of Liability

Scott Edwards Electronics, Inc. is not responsible for any special, incidental, or consequential damages resulting from any breach of warranty, or under any legal theory, including lost profits, downtime, goodwill, damage to or replacement of equipment or property, and any costs or recovering, reprogramming, or reproducing of data associated with the use of the hardware or software described herein.

Warranty, Return/Repair/Replacement Policy

Scott Edwards Electronics, Inc. warrants this product against defects in materials and workmanship for a period of 90 days. If you discover a defect, we will, at our option, repair, replace, or refund the purchase price. Return the product with a description of the problem. We will return your product or its replacement via standard shipping. Expedited shipping is available at the customer's expense.

• **NOTE:** Physically abusing the module, removing the LCD Serial Backpack® G12864 daughterboard from the LCD, or attempting to repair or modify the module or the daughterboard, voids this warranty.

Trademarks

Windows® is a registered trademark of Microsoft, Inc. BASIC Stamp® is a registered trademark of Parallax, Inc. Other trademarked names that may be mentioned in the body of this document are the property of their respective holders.

Connecting Power

The G12864 has a built-in power supply and voltage regulator. The input to this supply is J1, a 2.1-by-5.5mm power jack. Use this jack to connect a 6- to 8-VAC or 7- to 10-VDC power source capable of at least 100mA. Because the supply is designed for AC input, it will also accept DC inputs of either polarity (tip + or –). The onboard power supply regulates the input power to stable 5VDC.

You may also connect regulated 5VDC power directly to the pins of J3 marked +5V and GND. Make sure that your power supply is regulated to 5VDC ±0.25V and is capable of supplying up to 100mA.

Do not reverse +5V and GND when supplying 5VDC through J3. These connections are protected by a diode on the G12864 board. A reversed connection will look like a short-circuit to your power supply and may damage it. A sufficiently powerful supply connected backwards may destroy the diode and the display.

Do not exceed 5.5VDC into J3. Overvoltage will damage the G12864.

Do not connect anything to J1 when supplying 5VDC through J3.

Do not tap into the G12864's 5-volt supply to power other devices. When the backlight is turned on, the regulator is near its maximum continuous current rating.

Connecting Serial Input

The G12864 accepts serial data from RS-232 (±10-volt) or logic-level (+5-volt) sources. Most serial ports output to a DB9 male connector. Use a straight-through male-to-female DB9 cable such as the one included with the starter kit to connect J2, the G12864's serial port. Additional cables are available from Jameco (800-831-4242; pn 25700), Digi-Key (800-344-4539; pn AE1020-ND), and other computer suppliers. If your computer has a DB25 serial-port connector, use an off-the-shelf DB25-to-DB9 adapter.

The figure on the next page shows the wiring of the G12864's DB9 connector (J2). All handshaking connections are looped back (cross-connected so that they are disabled). The G12864 uses only the TxD (transmit data), RxD (receive data), and SG (signal ground) lines.

If your application does not require bidirectional communication, and your computer/controller doesn't have a DB9-style serial connector, you may connect serial output (TxD) and signal ground to the pins of J3 marked SER and GND, respectively.

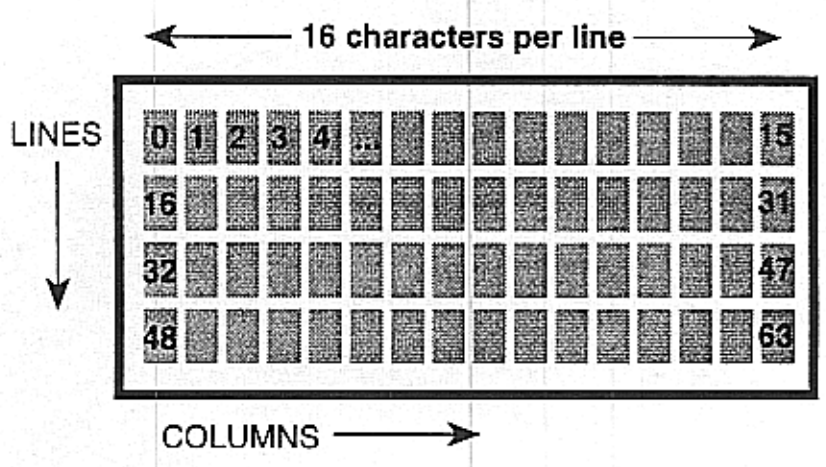
Getting Acquainted

The best way to get acquainted with the G12864 is to connect power and serial data as outlined in the previous sections, boot up a terminal communications program (like the Windows Terminal accessory or Procomm for DOS), and type some text and instructions. The text you type in your terminal software will appear on the display.

When you set up your terminal program, remember to

- Set the program and the display for the same baud rate.
- Configure the program for no parity, 8 data bits, 1 stop bit.
- Turn off hardware handshaking.
- Set the program for the COM port to which the display is connected.
- To see your typing on the PC monitor, set the program for "half duplex."

With the display powered and connected, anything you type in the terminal program will appear on the LCD. If it does not, check the settings and connection. If text is garbled, chances are that the baud rate is incorrect.



Layout of the 4x16 text layer.

As you send text to the display, characters print from left to right. When they reach the end of a line, the next character appears at the beginning of the next line. When text reaches the end of the display (character position 63) the next character appears at the top left position (character 0) of the display.

The display responds to the following special-purpose keys on your keyboard. In these descriptions, *cursor* means the position in which the next character typed will appear. The default mode is invisible cursor.

TAB: The cursor jumps to the next multiple-of-4 column position.

RETURN: The cursor jumps to the beginning of the next line.

BACKSPACE: The cursor backs up one column, erasing that location.

NULL

ASCII 0 *Control-@*

The G12864 ignores nulls without storing them in its data buffer. Sending a null is the equivalent of a brief time delay. Delay lengths depend on serial speed:

Speed (bps)	Null delay (ms)
2400	4.16
9600	1.04

CURSOR HOME

ASCII 1 *Control-A*

Control-A moves the cursor to position 0 (upper left corner) of the display.

IGNORED

ASCII 2 *Control-B*

Control-B is ignored but takes space in the buffer. Use control-@ if you need a time delay.

IGNORED

ASCII 3 *Control-C*

Control-C is ignored but takes space in the buffer. Use control-@ if you need a time delay. Many systems use control-C as a break character to halt execution; this will not affect the display.

BLANK CURSOR

ASCII 4 *Control-D*

Control-D makes the cursor invisible. If the cursor is already invisible, control-D has no effect.

V-BAR CURSOR ON

ASCII 5 *Control-E*

Control-E turns on a blinking cursor. The cursor is an eight-pixel vertical bar, similar to the insertion cursor of some text editors. To make the cursor invisible, send control-D.

V-BAR CURSOR ON

ASCII 6 *Control-F*

Same as control-E.

IGNORED

ASCII 7 *Control-G*

Control-G is ignored but takes space in the buffer. Use control-@ if you need a time delay.

BACKSPACE

ASCII 8 *Control-H* *(Bksp Key)*

Backspace, which may also be sent as control-H, causes the cursor to back up one column and print a space, leaving the cursor in that column.

One-byte binary method: Send the display position as a single byte value equal to the position plus 64. For example, from a terminal program, press control-P, then type "A". The cursor will jump to position 1 (second character of the first line) because the ASCII code for A is 65. The G12864 subtracts 64 from the binary value to arrive at the screen position.

With either method the G12864 will accept values larger than 63 (the highest valid position on a 4x16 screen). The cursor will wrap around to *position–64*. With the text method, it is possible (though not sensible) to send any decimal number as a position value. The number will be truncated to 8 bits *and* if larger than 63, wrapped around to *position–64*.

IGNORED

ASCII 17 *Control-Q*

Control-Q is ignored but takes space in the buffer. Use control-@ if you need a time delay.

BUFFER DEPTH PROBE

ASCII 18 *Control-R*

Control-R instructs the display to send the > character (ASCII 62). This permits an easy-to-implement form of serial flow control. Here's how it works: The G12864 has a 64-byte buffer that stores incoming serial data until the display is ready to process it. This is necessary because some instructions can arrive faster than the display can process them. For example, at 9600 baud, the clear-screen instruction arrives in 1.04 ms, but requires 15 ms to execute. While the screen is being cleared, the buffer accumulates data for later processing.

Many other instructions are processed much faster than they can arrive. This gives the buffer a chance to empty. However, if the input contains a large proportion of time-consuming instructions, it's possible for the buffer to overflow and lose incoming data. See the instruction-time table on page 23 for the execution times of the various instructions.

One way to prevent the buffer from overflowing is to ask the sender to stop transmitting while the display catches up. Unfortunately, that assumes that the sender can send and receive serial data simultaneously (*full-duplex*), and that it is capable of quickly shutting off its serial output. Many small controllers are incapable of full-duplex operation, and many large ones have such deep output buffers that they cannot stop quickly.

The buffer depth probe overcomes both problems. To use it, write your program so that it sends cntl-R (ASCII 18) before transmitting to the G12864. Wait for the > character (ASCII 62) to be returned. This indicates that the buffer is empty, so your program can now send up to 64 bytes without concern about buffer overflow.

Many applications will not require any flow control at all! The time required for your program to prepare data will usually give the display enough time to empty the buffer. For minor timing adjustments, consider sending an occasional null character (ASCII 0) or inserting a brief time delay in your program.

Graphics-Layer Instructions (Escape Sequences)

The table on the next page lists the graphics instructions. These are called escape sequences because they begin with the escape character (ASCII 27), which distinguishes them from normal text. If you set configuration switch 4 (ESC/CNTL-Z) to the CNTL-Z position, then cntl-Z (ASCII 26) substitutes for Escape. Leave this switch in the ESC position unless you are using terminal software that does strange things (changes screen colors, moves the cursor) in response to the escape sequences. Even if you're using cntl-Z, this manual still refers to them as *escape* sequences.

If escape sequences aren't working, disconnect the G12864 from power and move configuration switch 4 (ESC/CNTL-Z) to the ESC position (off/left). Reconnect power and try again.

All escape sequences follow the same basic pattern: *escape—letter—number*. For example, the escape sequence for displaying a screen stored in EEPROM is esc E*n*, where *n* is a number from 0 to 7. From the keyboard, you would press the Escape key, then E, then a number key like 2, then a non-numeric key like the space bar. As soon as you press the space bar, EEPROM screen 2 would load into the graphics layer of the display. The final, non-number character in the sequence, the space in our example, is thrown away; it does not affect the display except to signal the end of the number.

Some escape sequences take more than one number. You can use any non-numeric character other than null (ASCII 0) to separate the numbers. As we saw in the previous example, the last number must be followed by a non-numeric throwaway character.

Numeric entries are always sent as decimal text, like "123", or single-byte shortcuts.

Numbers up to 191 may be sent as one-byte shortcuts. Just add 64 to the number and send that single-byte value. For example, to send the number 101, you could either send the three bytes "101" or a single byte containing the value 64 + 101= 165 (i.e., A5 hex 10100101 binary). When you send multiple numbers as shortcuts, you do not need any characters to separate them, nor any throwaway character(s).

SET BYTE-WRITE (A)DDRESS

ESC A x y

Set the address for the next byte-write instruction (ESC B n). The x coordinate can range from 0 to 127; y from 0 to 7.

WRITE A (B)YTE

ESC B n

Write the byte value represented by n (0—255) to the current screen address, set by the instruction ESC A x y. Bytes are treated as vertical columns of bits with the low bit (lsb) at the top (page 18). Bits that are set to 1 show as black pixels; 0s as white. After this instruction, the write-byte address is automatically incremented by 1. If your program requires writing bytes to sequential screen locations (left to right and top to bottom) you need only to set the starting address.

(C)LEAR THE SCREEN

ESC C n

Write the byte value represented by n (0—255) to every byte address on the screen. In most cases, n is 0, which clears the screen to white. However, values of n from 1 to 254 may be used to produce striped patterns, or 255 to clear the screen to black.

(D)OWNLOAD A GRAPHIC

ESC D G

Download a bitmapped graphic to the graphics layer of the G12864. Note that the G in the command is literal; send a capital letter "G" (ASCII 71) in this position. After sending this instruction, a program should either wait for the G12864 to respond with the > character (ASCII 62) or pause without sending any further data for 500ms or more. The download process accepts binary data, so it temporarily suspends the G12864's usual behavior of discarding nulls (ASCII 0). Waiting for the > (or delaying for a half second) ensures that the download instruction is processed before any subsequent data is sent.

A total of 1024 bytes make up a graphic screen. These bytes map to the screen in exactly the same fashion as those sent via the write-byte instruction—working left to right and top to bottom. See the Painless Graphics Primer (page 18) for a map of the graphics screen.

The G12864 starter package includes a simple DOS utility for converting and downloading .BMP files.

SHOW (E)EPROM IMAGE

ESC E n

Transfer a 1024-byte image from flash EEPROM screen n to the graphics layer. Valid numbers are 0 through 15. Screens 0 and 1 are the text font.

SET (I)NK COLOR

ESC I n

Set the color of pixels used in subsequent plot and line instructions to n, where 0 is white and 1 is black.

DRAW LINE (T)O

ESC T x y

Draw a line from the last end coordinate specified in a line or line-to command to point x,y in the current ink color and drawing mode. Valid x coordinates range from 0—127 and y coordinates 0—63.

ENABLE LA(Y)ERS

ESC Y n

Selectively turn the text and/or graphics layers on/off according to the value of n, where 0 is both on, 1 is graphics off, 2 is text off, and 3 is both off.

Note that when one layer is off, the overlay logic for combining layers is ignored. For example, when the overlay logic is set to AND, text is visible only where it overlays black portions of the graphics screen. If the graphics screen is turned off, the entire text screen becomes visible. If the overlay logic remained in effect, the text screen would disappear—not particularly useful, since it duplicates the effect of ESC Y 3.

(X)FER —TRANSFER—GRAPHICS SCREEN TO EEPROM

ESC X n

Transfer the current graphics screen to flash EEPROM screen n, where n is in the range 0—15. This instruction may be disabled by setting configuration switch 5 to the PRTCT (write-protect) position. Transferring data to EEPROM is a time-consuming process, so the G12864 sends a > character (ASCII 62) when done. The > is sent even if the transfer is locked out by the PRTCT setting.

Use caution with XFERS since EEPROM screens 0 and 1 contain the font bitmaps used on the display's text layer. If you accidentally overwrite these images, text may change to hieroglyphics! You will have to restore the fonts from the included starter disk.

OVERLAY LOGIC

The G12864 stores text and graphics in separate memory layers, then combines them for display on the screen. You can set the way that the layers are combined to achieve various visual effects. Here are the three options:

OR Black text shows only when it overlays a white portion of the graphics layer.

XOR Black text shows as the inverse of the graphics it overlays.

AND Black text shows only when it overlays a black portion of the graphics layer.

To understand these options more completely, let's look at how individual pixels from each layer would combine with each of the logic options.

text	graphics	logic	screen	remarks
▒	▒	OR	▒	0 OR 0 = 0
▒	■	OR	■	0 OR 1 = 1
■	▒	OR	■	1 OR 0 = 1
■	■	OR	■	1 OR 1 = 1
▒	▒	XOR	▒	0 XOR 0 = 0
▒	■	XOR	■	0 XOR 1 = 1
■	▒	XOR	■	1 XOR 0 = 1
■	■	XOR	▒	1 XOR 1 = 0
▒	▒	AND	▒	0 AND 0 = 0
▒	■	AND	▒	0 AND 1 = 0
■	▒	AND	▒	1 AND 0 = 0
■	■	AND	■	1 AND 1 = 1

USING THE DOWNLOADING PROGRAM

The conversion/downloading program is a simple DOS program. If you are running Windows 3.1 or 95, you must go to the DOS prompt. There are several ways to do this:

Win 3.1	• Double-click "MS-DOS Prompt" from within Program Manager. • Close the Program Manager window to "end your Windows session."
Win 95	• From the Start Menu, choose Programs, then MS-DOS prompt • From the Start Menu, choose Shutdown and pick "Restart the computer in MS-DOS mode."

If the program doesn't seem to work correctly in a DOS box (the first choice in the table), you can try it in pure DOS mode (the second choice).

To use the downloading software, copy the file BMPX.EXE to your hard drive. A good location is the directory (folder) named DOS, since this is where other DOS-related program and command files reside. A good second choice would be in the same directory as the .BMP file(s) that you plan to work with.

Connect the G12864 to a PC serial port (COM1 or COM2) and power it up. To download a .BMP file, get to the DOS prompt and type:

```
BMPX COM1 9600 filename{.BMP}
```

Substitute `COM2` if you are using that port, and/or `2400` if your G12864 is set for that speed. If the file is in the current directory, then you may use just the file name. If it's in a different directory, you must provide the complete path (e.g., `C:\PAINT\MY_FONT1`). Note that if the filename ends in .BMP, you may omit the file extension when typing the command.

The program will convert the bitmap file to the correct format for the G12864, then download it to the display. The program will then ask whether you want to transfer the picture to flash EEPROM, and if so, to which location (0—15). If you select 0 or 1, the program will verify that you really want to overwrite the font memory.

The display will not transfer a picture to flash EEPROM if configuration switch 5 is set to PRTCT (write-protect). Turn off the display and change the switch setting before trying again. The software does *not* provide any error message when this occurs.

Design Notes and Specifications

The G12864 is designed to speed prototyping and enable the creation of bench-top instruments requiring a serially driven text/graphics display. It is not meant for direct use by non-technical users or for incorporation into mass-produced consumer equipment. It has not been tested to EMC/EMI/ESD standards. (The same is true of most single-board computers and other subassemblies whose specs do not specifically include compliance data.) It is up to the purchaser to determine whether the G12864 is suitable for the intended use.

This unit is designed and manufactured to high standards of quality. This notice is simply intended to head off invalid assumptions that might affect decisions about using the G12864 in consumer products.

MAXIMUM RATINGS AND ELECTRICAL CHARACTERISTICS

The following data is preliminary and subject to change without notice. The G12864 consists of two subassemblies—an LCD graphics module, and a custom serial controller. Where the ratings of these two devices vary, the more conservative rating is shown in the table.

Parameter	min	max	typical	unit
Operating temperature range	0	+50	—	°C
	+32	+122	—	°F
Storage temperature range	–20	+70	—	°C
	–4	+158	—	°F
Supply voltage to J1 (Vin)	7.0	10.0	9.0	Vdc
	6.0	8.0	7.0	Vac
Supply voltage to J2 (Vsupply)	4.50	5.50	5.00	Vdc
Current draw, backlight off	—	25	15	mA
Current draw, backlight on	—	100	75	mA
Input voltage, serial 1	–15	+0.5	—	V
Input voltage, serial 0	3	15	—	V
Output voltage, serial 1	–9	0	–5	V
Output voltage, serial 0	3	Vsupply	4.5	V
Processor clock speed	—	—	16	MHz

MAXIMUM EXECUTION TIMES FOR LCD INSTRUCTIONS

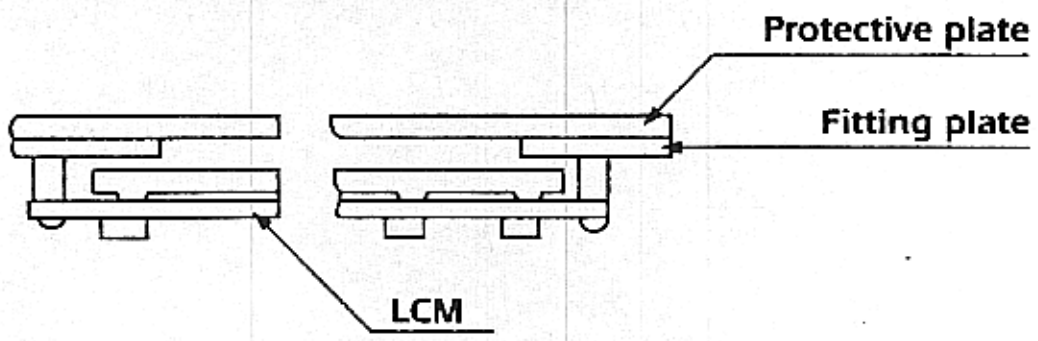
The following data is preliminary and subject to change without notice. Instructions not listed in this table take less than 1 ms and therefore do not cause data to accumulate in the serial input buffer. The buffer can hold a maximum of 64 bytes. All times are measured from entry of the last instruction parameter to the processing of the next instruction.

Instruction/action	max	units
Display a text character (ASCII 32—159)	1.2	ms
Backspace (ASCII 8)	1.1	ms
Clear text (cntl-L)	15	ms
Clear graphics (ESC C)	15	ms
EEPROM screen (ESC E)	55	ms
Line plot (ESC L or ESC T)	55	ms
Transfer a screen to EEPROM (ESC X)	150	ms

- Don't allow the LCD to come into contact with oil and fats.
- Condensation on the surface and contact terminals due to cold will damage, stain or dirty the polarizers. After products are exposed to low temperatures they must be allowed to warm up before coming is contact with room-temperature air.
- Do not attach anything to the display screen. This may leave scratches or marks.
- Do not touch the display screen with bare hands. This will stain it. Skin oils and cosmetics are detrimental to the polarizers.
- Do not drop or jar the LCD module as you may crack or chip the glass.

INSTALLING THE LCD MODULE

The LCD manufacturer suggests the mounting arrangement shown below. The *protective plate* would be a transparent window; the *fitting plate* would be the panel or enclosure. The standoff posts are threaded for 2/56 machine screws.



RECIPE 2: DISPLAYING AN EEPROM SCREEN

This recipe shows how to display an image stored in EEPROM and flash the image between normal and reversed modes. It assumes that the factory-installed screens are still stored in memory. If they are not, the images displayed will vary from the description.

Type this:	Purpose/effect
`<cntl-L>`	Clear text screen
`<esc> E 2 <space>`	Show EEPROM image 2 (G12864 title screen)
`<esc> E 6 <space>`	Show EEPROM image 6 ("Intruder Alert")
`<esc> R 1 <space>`	Reverse (invert) the graphics screen
`<esc> R 0 <space>`	Un-reverse the graphics screen

RECIPE 3: COMBINING TEXT AND GRAPHICS

This recipe shows how to use a graphics-layer image to highlight text. It assumes that the factory-installed screens are still stored in memory. If they are not, the images displayed will vary from the description.

Note: the O in <esc> O 1 <space> is the letter O. Remember to use <cntl-Z> instead of <esc> if your G12864 is set for <cntl-Z>.

Type this:	Purpose/effect
`<cntl-L>`	Clear text screen
`<esc> C 0 <space>`	Clear graphics screen to 0 (zero, blank)
`<space> MENU ITEM 1 <return>`	Text
`<space> MENU ITEM 2 <return>`	Text
`<space> MENU ITEM 3 <return>`	Text
`<space> MENU ITEM 4 <return>`	Text
`<esc> O 1 <space>`	Set overlay logic to XOR.
`<esc> E 8 <space>`	Show image 8 (Highlight bar, line 1)
`<esc> E 9 <space>`	Show image 9 (Highlight bar, line 2)
`<esc> E 10 <space>`	Show image 10 (Highlight bar, line 3)
`<esc> E 11 <space>`	Show image 11 (Highlight bar, line 4)

EXAMPLE PROGRAM 1: PC RUNNING QBASIC OR QUICKBASIC

```
' Program: QGXDEMO.BAS (QBASIC/QuickBASIC Demo of G12864)
' This program demonstrates the G12864 Serial Graphics LCD
' from Scott Edwards Electronics Inc. with QBASIC or QuickBASIC.
' Configure the G12864 for 9600 baud (config switch 2 ON) and
' escape codes (config switch 4 OFF). Connect power and serial
' input. This program uses COM1--change the "OPEN" instruction if
' you're using COM2.

'================Delay Subroutine using Nulls====================
DECLARE SUB pause (time AS INTEGER)
DEFINT A-Z

'===========Constants Defining LCD Instructions================
' These constants assign names to the LCD instructions. The
' numbers are the ASCII values listed in the manual. For example,
' cntl-L is the clear-screen instruction; the ASCII value is 12.
' So printing CHR$(12) to the LCD clears the screen.
CONST clrTXT = 12          ' Cntl-L: clear the text layer.
CONST bksp = 8             ' Backspace.
CONST posCmd = 16          ' Position the cursor.
CONST ESC = 27             ' Escape code.

'==================Demonstration Program======================
' Set up serial port COM1; disable handshaking (CD, CS, DS, OP).
OPEN "COM1:9600,N,8,1,CD0,CS0,DS0,OP0" FOR OUTPUT AS #1
' ===========
' ==Recipe 1
' ===========
' This code works the same as recipe 1 from the G12864 manual.
' It clears the screen, prints some text, then positions the
' cursor to change just portions of the text. Notice that some
' PRINT statements end in semicolons (;) and others don't. PRINT
' normally adds a carriage return to all data. When you don't want
' a return (as when you send instructions), adding a semicolon
' prevents it.
PRINT #1, CHR$(clrTXT);          ' Clear text layer.
PRINT #1, CHR$(ESC); "C0.";      ' Clear graphics layer.
PRINT #1, "RECIPE NUMBER 1"      ' Print the title, followed by return.
PRINT #1, "Power:"               ' Print label, followed by return.
PRINT #1, "Speed:"               ' "      "
' Here's the cursor-positioning command. This PRINT statement combines
' the command with the text to be printed. Notice that there's a
' throwaway character (.) separating the number (23) from the text "75".
' You could use any character except null (ASCII 0) in that position.
PRINT #1, CHR$(posCmd); "23.75%";        ' Go to position 23, print "75%"
PRINT #1, CHR$(posCmd); "39.87mph";      ' Go to position 39, print "87mph"
SLEEP 2                                  ' Delay for 2 seconds.
```

```
' ===========
' ==Recipe 3
' ===========
' Recipe 3 shows a useful technique for combining text and graphics to
' selectively highlight portions of the text.
PRINT #1, CHR$(clrTXT);          ' Clear text layer.
PRINT #1, CHR$(ESC); "C0.";      ' Clear graphics layer.
PRINT #1, " Menu Item 1"         ' Display text, followed by return.
PRINT #1, " Menu Item 2"         ' "
PRINT #1, " Menu Item 3"         ' "
PRINT #1, " Menu Item 4"         ' "
PRINT #1, CHR$(ESC); "O1.";      ' Set (O)verlay to 1 (XOR).
FOR selection = 8 TO 11          ' EEPROM screens 8-11 are highlight bars.
  SLEEP 1                        ' Pause 1 second.
  hilite = selection + 64        ' One-byte shortcut for selection.
  PRINT #1, CHR$(ESC); "E"; CHR$(hilite);        ' Show selection screen.
NEXT
SLEEP 2                          ' Pause 2 seconds.
' ===========
' ==Recipe 4
' ===========
' This code draws a box on the screen with the line and line-to
' commands. It's a little different from the recipe in the manual
' in that it uses only shortcut numbers instead of numeric text.
PRINT #1, CHR$(clrTXT);          ' Clear text layer.
PRINT #1, CHR$(ESC); "C0.";      ' Clear graphics layer.
x1 = 10 + 64: y1 = 10 + 64       ' 1st point (64 added to make shortcuts)
x2 = 117 + 64: y2 = 10 + 64      ' 2nd point.
' Here's the line command in the form <esc> L x1 y1 x2 y2
PRINT #1, CHR$(ESC); "L"; CHR$(x1); CHR$(y1);
PRINT #1, CHR$(x2); CHR$(y2);
' Now we can use line-to commands. This draws a line that starts
' from the endpoint of the previous line.
y2 = 53 + 64                     ' New y coordinate
PRINT #1, CHR$(ESC); "T"; CHR$(x2); CHR$(y2);
x2 = 10 + 64                     ' New x coordinate
PRINT #1, CHR$(ESC); "T"; CHR$(x2); CHR$(y2);
y2 = 10 + 64                     ' New y coordinate
PRINT #1, CHR$(ESC); "T"; CHR$(x2); CHR$(y2);
SLEEP 2
```

EXAMPLE PROGRAM 2: BASIC STAMP II

```
' Program: GXDEMO.BS2 (Demonstrate G12864 Serial Graphics LCD)
' This program demonstrates many of the features of the G12864
' Serial Graphics LCD Module from Scott Edwards Electronics.
' Set the module to 9600 bps (config switch 2 ON/right).
' Make sure that ESC/ctl-Z is set to ESC (config 4 OFF/left).
' Connect the serial input of the module (J3 pin marked SER)
' to BS2 pin P0, GND to GND and +5V to +5V. Alternatively, you can
' omit the +5V connection and run the G12864 off its AC adapter.
' ======IMPORTANT====== If you run the G12864 off the BS2's 5V
' regulator, do not turn on the backlight. The BS2 regulator
' cannot supply enough current to power it.

' =========Define names for LCD instructions, bps=======
LCD     con     0       ' Output pin to LCD.
clrLCD  con     12      ' Clear LCD text screen.
posCmd  con     16      ' Position cursor.
N9600   con     $4054   ' Baudmode for inverted, 9600-bps output.
N2400   con     $418d   ' Baudmode for inverted, 2400-bps.
ESC     con     27      ' Escape character.
' Note: the constant "CR" for carriage return is predefined in the BS2.

' ====================Define variables==================
counter var     byte    ' Byte counter for miscellaneous purposes.
sinX    var     byte    ' Sine of x used in sine-plot demo.

' ===========Recipe #1, positioning the cursor==========
pause 1000
serout LCD,N9600,[ESC,"C0 "]                    ' Clear graphics screen
serout LCD,N9600,[clrLCD,"RECIPE NUMBER 1",cr]  ' Clear screen, show text.
serout LCD,N9600,["Power:",cr,"Speed:"]         ' More text.
' The following lines position the cursor by sending the positioning
' command, followed by a decimal number (e.g., DEC 23) and a space
' (throwaway character). This is the same way you would type the command.
serout LCD,N9600,[posCmd,DEC 23," 65%"]         ' Move to pos 23, print 65%
serout LCD,N9600,[posCmd,DEC 39," 70mph"]       ' Pos 39, print 70mph
' Within programs, there's a better way to send numbers--just add 64
' to the desired value and send that as a single byte. Since a byte
' can hold up to 255, you can send any value from 0 to 191 with this
' technique.
for counter = 70 to 100 step 5
  pause 1000                                    ' Wait a second.
  serout LCD,N9600,[posCmd,(64+23)]             ' Move to pos 23 by 1-byte method.
  serout LCD,N9600,[DEC counter,"%"]            ' Print the count there.
  serout LCD,N9600,[posCmd,(64+39), DEC (counter+5),"mph"]  ' Pos 39, text.
next
pause 2000                                      ' Wait 2 seconds.
serout LCD,N9600,[clrLCD]                       ' Clear text screen.
```

```
' ================Bonus, Plotting a sinewave==============
pause 2000                          ' Wait 2 secs.
serout LCD,N9600,[ESC,"C",64]       ' Clear graphics screen.
' The loop below displays several cycles of a sine wave using the
' BS2's SIN function. It plots the wave centered vertically on the
' screen (above and below y coord 32).
for counter = 0 to 127
  sinX = 32+((SIN (8*counter))/8)
  serout LCD,N9600,[ESC,"P",(64+counter),(64+sinX)]
next
stop                                ' Demo over.
```